Санкт-Петербургский государственный университет

Фундаментальная информатика и информационные технологии
Автоматизация научных исследований

Чиликин Александр Сергеевич

Построение контура объекта по набору изображений

Магистерская диссертация

Научный руководитель:
к. ф.-м. н., доцент Погожев С. В.

Санкт-Петербург
2018

Оглавление

Введение

Вместе с бурным развитием вычислительной техники в 21 веке появились или также получили развитие различные области науки и инженерного искусства. Еще всего лишь несколько десятилетий назад едва ли были мыслимы роботы Boston Dynamics, эксперименты ЦЕРН на БАК, современные компьютерные игры, интеллектуальные системы типа Siri от Apple, в основе которых лежит обработка очень больших объемов данных и применение state-of-art алгоритмов машинного обучения и нейронных сетей, программное обеспечение, умение владеть которым сегодня является скорее обыденностью и необходимостью, нежели преимуществом, — взять те же графические редакторы, и т. д.

Одной из таких областей является компьютерное зрение — междисциплинарная область, появившаяся в последней четверти прошлого века и занимающаяся извлечением высокоуровневых представлений из цифровых изображений и видео, а также автоматизацией задач, которые выполняет обычный человеческий глаз. При этом медиаисточники, в соответствии с решаемой задачей, могут иметь иметь самый разный характер и содержание. Рассмотрим несколько примеров.

Главной областью, где результаты компьютерного зрения были востребованы, является робототехника. Первые системы компьютерного зрения были связаны с промышленными роботами и проверяли, что деталь на конвейере находится в определенном месте и занимает определенное количество пикселей. Сейчас появляются роботы, которым надо работать не с деталями, лежащими на плоском конвейере с известным цветом ленты, а от которых требуется способность понимать, анализировать реальный мир в трехмерном пространстве, движения и намерения объектов в нем, и это на порядки сложнее.

Элементы компьютерного зрения также используются в стоматологии. Ранее для создания коронок копировали форму обломка зуба, создавая слепок, по которому изготавливали новый зуб. При помощи современных технологий можно обойтись без слепков — необходимо взять специальную камеру, снять обломанный зуб со всех сторон и по-

строить трехмерную модель обломка. Затем на основе совмещения и вычитания объемов этой модели и модели целого зуба получется модель для заготовки, которая уже отправляется на печать. Также есть примеры диагностического применения. По срезам МРТ головного мозга можно получить трехмерную картинку, из которой будет понятно, какого объема тот или иной процесс в органах пациента, насколько сильно он распространен, что поможет врачу лучше проанализировать ситуацию.

Беспилотным автомобилям необходима способность находить людей, светофоры и их сигналы, чтобы избегать ДТП, распознавать разметку, топографические и искусственные объекты с картографическими данными, чтобы удерживаться в своей полосе, контролировать своё местоположение и корректировать его при необходимости.

В ритейле компьютерное зрение применяется для распознавания штрих-кодов и подсчета посетителей.

Другим следствием развития вычислительной техники было появление относительно дешевых, простых в использовании и обладающих малыми размерами, программируемых контроллеров. Одним из таких контроллеров является линейка «аппаратно-вычислительных платформ» Arduino. Благодаря ей многие познали мир микроконтроллеров, узнали, что собрать небольшое полезное устройство может даже человек без специального образования, с минимальными познаниями в программировании и с отсутствием познаний в электронике — доказательством тому служит множество образовательных проектов по робототехнике, ориентированных на школьников. Однако это вовсе не означает, что данные контроллеры не применяются в серьезных проектах.

Данная работа представляет собой проект устройства, предназначенного для решения конкретной задачи построения контура объектов по набору изображений в рамках создания станка с ЧПУ с использованием контроллера Arduino в аппаратной части и алгоритмов компьютерного зрения в программной.

Постановка задачи

Рассмотрим задачу производства различных изделий из ткани. Пусть имеется (штучное) производство футболок с полной запечаткой и пошивом, а также иных изделий из ткани. Одним из этапов этого производства является вырезание элементов одежды из куска ткани, на котором производилась печать. При промышленных масштабах это производится на широкоформатных режущих плоттерах с использованием заранее заданного шаблона. При штучном производстве это делают руками с помощью обычных ножниц. Еще один из способов достичь цели — с помощью лазера. При этом суть метода остается такой же, как в варианте с плоттером, однако на месте режущих ножей будет стоять лазер. Перечислим основные плюсы и минусы данного подхода по сравнению с остальными.

Плюсы:

1. Автоматизация ручного труда.
2. Не происходит растяжения ткани и, кроме того, она оплавляется по краям.
3. Возможность использования лазера для резки других материалов и изделий.

Минусы:

1. Повышенные требования к технике безопасности.
2. Стоимость данного варианта сильно превышает стоимость ручного труда.
3. Необходимость наличия специального помещения или стола с вытяжкой.

Сделаем еще несколько замечаний. Во-первых, печать на ткани подвержена многим факторам, и в общем случае каждый раз результат печати по одному и тому же шаблону будет немного отличаться от

предыдущего. Поэтому при резке на традиционном плоттере могут возникнуть проблемы (вдобавок к тому, что плоттер сам по себе весьма дорогой инструмент, и его применение нецелесообразно при штучном производстве). Во-вторых, еще раз стоит отметить, что при резке ножницами ткань может неравномерно растягиваться, что негативно сказывается на качестве итогового результата (изделия).

В условиях наличия в распоряжении части аппартных средств для создания собственного станка-резака было предложено спроектировать и реализовать его. В самом общем виде техническое задание звучало следующим образом: из куска ткани (пример на рис. 1) с напечатанными на ней частями одежды нужно вырезать эти самые части. При этом печать всегда происходит на белой ткани и на границах вырезаемых элементов есть тонкие (но заметные) контрастные контура (иначе без шаблона было бы невозможно понять, что нужно вырезать, если часть элемента по задумке должна остаться белой). Резку в силу обозначенных выше причин и замечаний было предложено производить лазером, а определение элементов осуществлять при помощи цифровой камеры. Допустимая погрешность была определена в 1-2 мм. Критерием корректности резки является оценка экспертом (дизайнером). Таким образом, более конкретно задачу можно сформулировать так: необходимо *создать станок для лазерного раскроя швейных материалов, оснащенный интеллектуальной оптической системой*. При этом критерий корректности его работы математически выражается следующим образом: $\forall E \in \mathcal{A}\ \forall p \in E\ \exists q \in L_E$: $||p - q|| <= 2$ мм, где $\mathcal{A}$ – множество контуров вырезаемых элементов, E – множество точек контура вырезаемого элемента, L – множество точек, на которые попал луч лазера при вырезании соответствующего контура E.

Существующие решения

Основными параметрами, которые будут рассматриваться при рассмотрении существующих решений, являются:

1. Размеры рабочей области и возможность ее увеличения
2. Наличие камеры, предназначенной для формирования управления станком
3. Возможность (заявленная производителем или автором решения) раскроя ткани.
4. Цена

Лазерная резка на сегодняшний день является очень высокоэффективным способом обработки материалов различного типа, что обусловливает ее большую популярность. В связи с этим можно ожидать, что на рынке имеется множество предложений самого различного вида, а также существуют любительские решения, в том числе с готовым программным обеспечением с открытым исходным кодом.

Действительно, лазерных станков на рынке очень много. Однако полностью подходящих под указанные параметры найти не удалось. Некоторые станки используют *аналоговые* CCD-камеры, установленные прямо на голову с лазером.

Они работают следующим образом. На рабочем столе лазерного станка размещается материал с нанесенным на него (вышитым, нарисованным, награвированным и т.д.) изображением (множеством идентичных либо различных рисунков). Встроенная цифровая камера, расположенная рядом с лазерной головой, сканирует изображенные на материале объекты и передает их на экран монитора компьютера. Оператор выбирает необходимый ему объект и устанавливает (корректирует) параметры резки. Далее, при запуске процесса, лазерный луч вырезает все объекты, соответствующие заданным ранее контурам изображения, используя при этом параметры резки, установленные оператором

лазерного станка. Те объекты на материале, которые не соответствуют заданным контурам изображения, останутся нетронутыми.

Например, станками с таким устройством являются CCD IL-6090 SGC от компании Interlaser, ETS Series Laser Cutting Machine with CCD Camera System от компании Dongguan Eastern Laser Tech Co., Ltd., MT-9060CF от компании Dongguan Mactron Technology Co., Ltd, и многие другие.

Стоит отметить систему i-Cut, используемую на станках фирмы Trotec, которая предполагает использование специальных меток на вырезаемом материале.

Интересными являются машины от Shanghai KASU Intelligent Technology Co., Ltd. У них имеются машины, проецирующие изображение на рабочую область. Уже после этого оператор кладет материал на рабочую область и происходит работа. На другие их машины установлены SRL-камеры высокого разрешения и имеется полноценная визуальная система распознавания. Они работают по принципу, очень схожему предлагаемому в данной работе.

Однако все эти машины обладают следующими недостатками:

1. Высокая цена, начинающаяся в среднем от 500-700 тысяч рублей (там где указана)

2. Это полноценные решения, ориентированные на производство в промышленных или близких к ним масштабах.

3. Закрытое программное обеспечение и отсутствие возможностей для кастомизации.

Эти недостатки обусловливают ценность данной работы по проектированию простой, но дешевой и расширяемой оптической системы для лазерного станка.

Глава 1. Обзор и аппаратная часть реализации

Спроектированный станок имеет вид, показанный на рисунке 2.

Технически он представляет собой обычный станок с ЧПУ, работающий под управлением контроллера Arduino Uno на прошивке Grbl. Непосредственный элемент управления, часть станка, именуемая головой, может перемещаться по двух перпендикулярным осям. Область, по которой она перемещается, имеет строго заданные ограниченные размеры, и в дальнейшем будет называться рабочей областью. Над рабочей областью на жестком штативе закреплена цифровая камера.

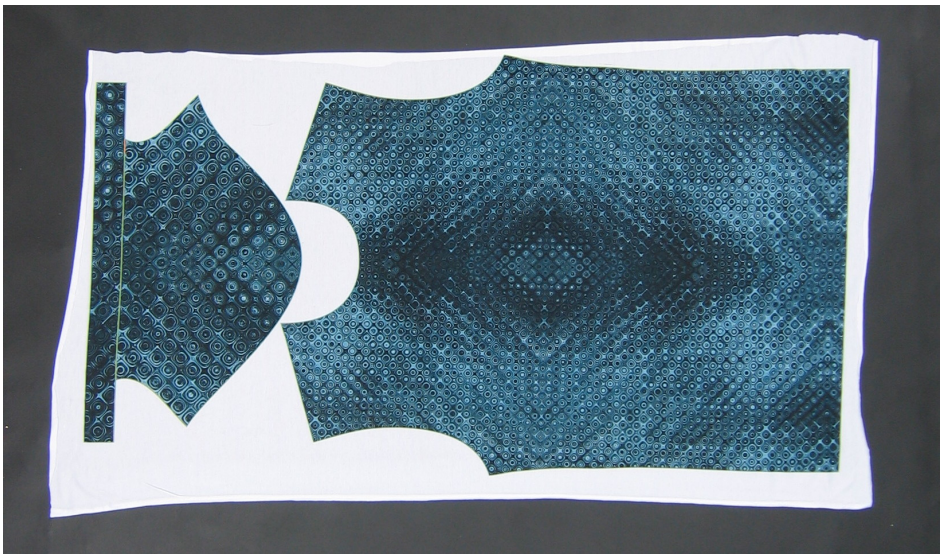

Рис. 1: Пример ткани с печатью

Станок работает следующим образом:

1. Сначала на рабочую область кладется ткань с напечатанными на ней изображениями.

2. Затем камерой производится съемка рабочей области.

3. Полученное фото передается программе, которая в рамках исходных предположений находит на изображении то, что нужно вырезать.

Рис. 2: Станок

4. По найденным на фото объектам формируется соответствующее числовое управление, обеспечивающее вырезание объектов.

В процессе проектирования и реализации станка оказались задействованы следующие материалы:

1. рамы из алюминиевых профилей;
2. шаговые двигатели Leadshine 57HS13 в количестве трех штук (1 на ось X, два на ось Y) [21];
3. два драйвера шагового двигателя Leadshine DM556 (по одному на ось) [22];
4. блок питания Leadshine SPS407, выходное напряжение 42 V, ток нагрузки 7 A, мощность 300 Вт [23];

5. микроконтроллер Arduino Uno [3];

6. необходимая электрическая обвязка;

7. штатив для камеры из алюминиевых профилей;

8. цифровая камера Canon PowerShot S50, разрешение 5.3 мегапикселя [24];

9. лазерная указка;

10. лазер твердотельный, длина волны 450нм, мощность 5 Вт [1].

Лазерная указка использовалась при разработке алгоритма поиска вырезаемых элементов.

Камера выбиралась исходя из следующих требований:

1. Она должна обеспечивать минимально необходимую точность, т. е. обладать таким разрешением, чтобы на один миллиметр рабочей области приходился как минимум один пиксель. Рабочая область имеет размеры 1380x1170 мм, поэтому минимально допустимое разрешение камеры 1380*1770 = 1.7 мегапикселя.

2. Должна иметься возможность управлять ею программно, т. е. при помощи кода изменять ее внутренние параметры, заставлять делать снимки и получать их.

3. Должна иметься возможность горизонтально закрепить ее на штатив.

Также для камеры было разработано решение стационарного электропитания вместо аккумуляторного.

Глава 2. Программная часть реализации

В начале работы программы требуется сфотографировать рабочую область. Для связи с фотографирующим устройством используется библиотека libgphoto2 [31], представляющая интерфейс к большому числу современных цифровых камер. Связь происходит по специально созданному для этой и других целей протоколу PTP (Picture Transfer Protocol).

Большинство объективов, особенно низкой ценовой категории, характеризуются присутствием всевозможных аберраций, то есть оптическими искажениями на снимке, которые напрямую связаны с системой оптики. Эти аберрации по своему происхождению могут быть хроматическими или геометрическими. Хроматические характеризуются появлением ненужных цветных ореолов и контуров на границах объектов. Однако применительно к поставленной задаче наибольший интерес вызывают геометрические аберрации, которые искажают геометрию снимаемых объектов и также называются дисторсией. Основные виды дисторсий приведены на рисунке 3.

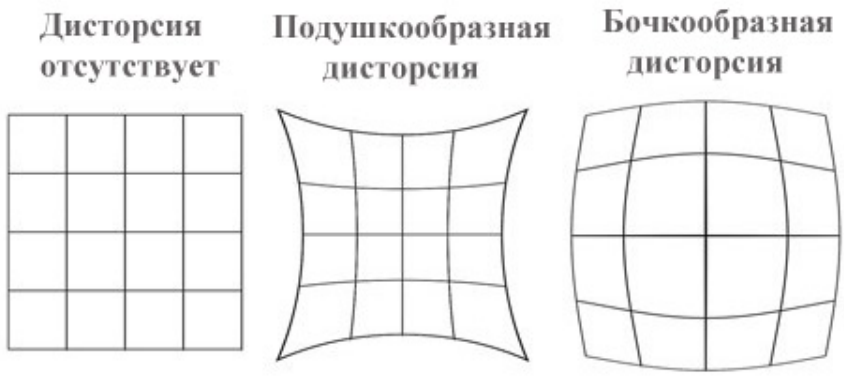

Рис. 3: Виды дисторсий

Для получения высокоточного результата требуется устранить эти дисторсии. В научной литературе эта тема является широко исследованной — сервис Google Scholar по запросу «lens distortion models» выдает 183 тысячи результатов. Исторически первой моделью дисторсий была «decentering distortion model», созданная в 1919 году немецким ученым Conrady. Примерно через 50 лет эта модель была рас-

ширена Brown [5, 6], который представил параметрическую «radial, decentring and prism distortion model», которая затем широко использовалась для представления слабых дисторсий [29, 19, 30, 11]. В дальнейшем предпринимались попытки математического уточнения этой модели [7, 2, 26, 27, 20, 18], а также были представлены и непараметрические модели, например [27]. Fitzgibbon [12] модифицировал «radial distortion model», предложив использование «division model», которая способна представить сильные дисторсии при меньшем числе параметров. В частности, при использовании этой модели для многих камер оказывается достаточно всего лишь одного параметра. Помимо перечисленного, существуют другие весьма специфические виды дисторсий и модели, их описывающие. Существуют также работы, проводящие сравнение существующих моделей, например [25].

Существует различное интерактивное программное обеспечение, позволяющее устранять дистории, которое использует специально созданные производителями профили объективов, содержащие посчитанные параметры тех или иных моделей. Однако их подавляющее большинство работает только с RAW-файлами (формат цифровых файлов изображения, содержащий необработанные данные об электрических сигналах с фотоматрицы), а также их невозможно использовать в автоматическом процессе.

Поэтому в данной работе рассмотрим следующую модель, базирующуюся на [6, 13] (также см. рис. 4), поскольку она является одной из самых простых, наиболее исследованных и в этой задаче показала результат, сравнимый с более сложными:

$$[x, y, z]^T = R\,[X, Y, Z]^T + t$$
$$x' = x/z, y' = y/z$$
$$x'' = x'\frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 x' y' + p_2(r^2 + 2x'^2)$$
$$y'' = y'\frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1(r^2 + 2y'^2) + 2p_2 x' y'$$
где $\quad r^2 = x'^2 + y'^2$
$$u = f_x * x'' + c_x, v = f_y * y'' + c_y$$

Здесь $k_1, k_2, k_3, k_4, k_5, k_6$ – коэффициенты радиальной дисторсии, p_1, p_2 – коэффициенты тангенциальной дисторсии.

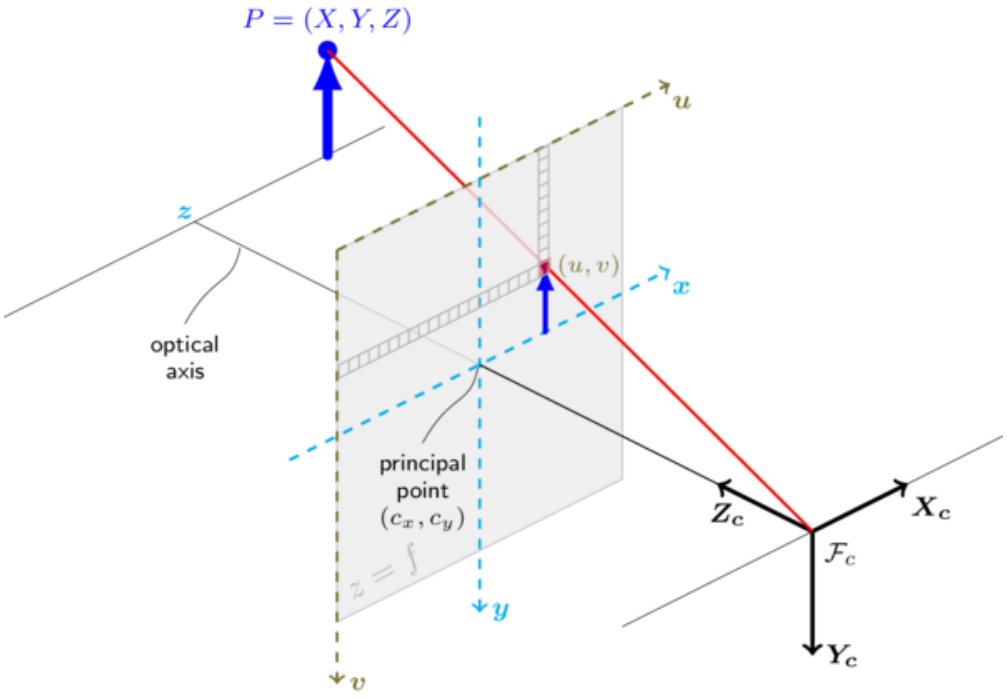

Рис. 4: Модель камеры-обскуры

Для того чтобы можно было устранить дисторсии, требуется определить обозначенные коэффициенты. Это можно сделать в процессе калибровки камеры. На самом деле при этом мы получаем не только указанные коэффициенты, но нас интересуют только они. Для этого требуется один и тот же предмет сфотографировать с разных ракурсов (рис. 5) и определить соответствие между точками в системе координат,

связанной с этим предметом, и точками на изображении. Для простоты используют плоские объекты с простыми паттернами. Стоит отметить, что калибровку нужно произвести всего лишь один раз, и снова определять полученные коэффициенты далее уже не требуется.

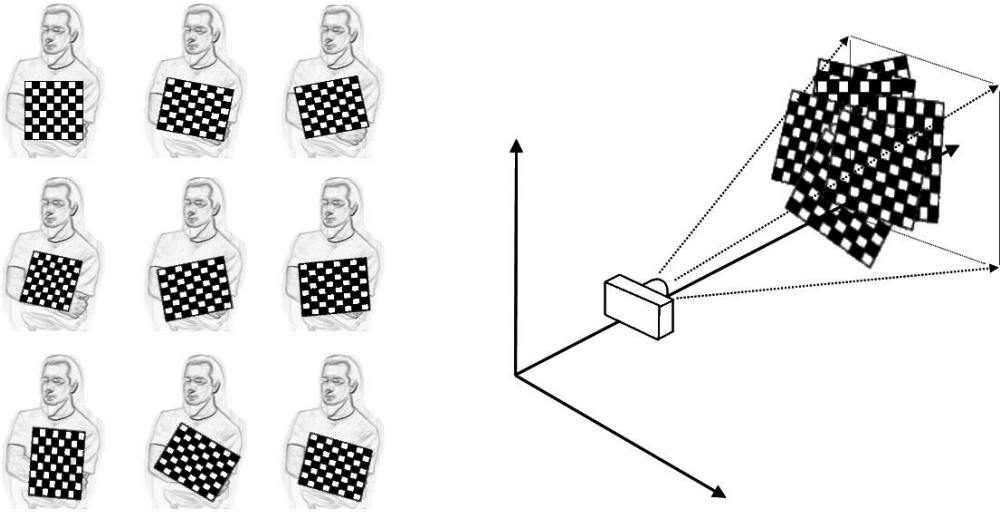

Рис. 5: Паттерн, сфотографированный с разных ракурсов

В данном работе предлагается использование паттернов ChAruCo (рис. 7), являющихся комбинацией шахматного паттерна и AruCo [4, 15], поскольку они также используются в этой работе для другой цели, а также потому что они обладают тем свойством, что алгоритм будет корректно работать в случае, если на фотографии находятся не все точки паттерна или не все их удалось распознать. Эти метки являются одной из разновидностей (рис. 6) т. н. fiducial markers — специальных объектов, используемых для определения положения объектов, и главным образом используемых сегодня в робототехнике и технологиях дополненной реальности.

Для нахождения точек паттерна и определения коэффициентов использовались алгоритмы, описанные в [4, 30, 8].

Оригинальное изображение с камеры и результат устранения дисторсий можно видеть на рис. 8 (обратите внимание на белую резиновую ленту у правого края изображений).

В силу влияния факторов различной природы, в первую очередь человеческих, задача установки камеры строго посередине рабочей области является нетривиальной. Даже будучи закрепленной на штативе,

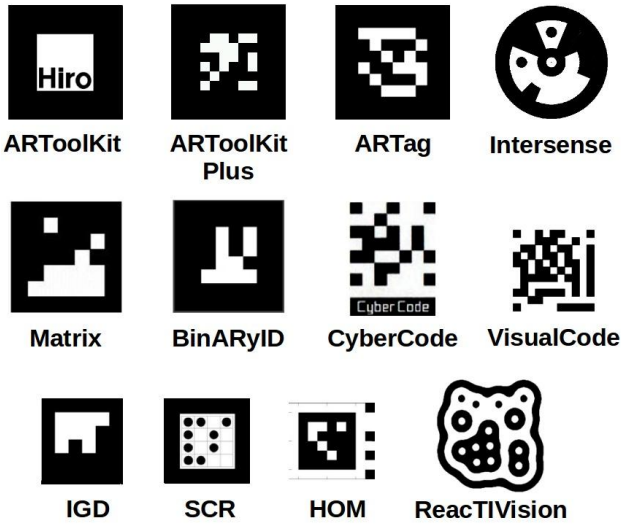

Рис. 6: Разновидности fiducial markers

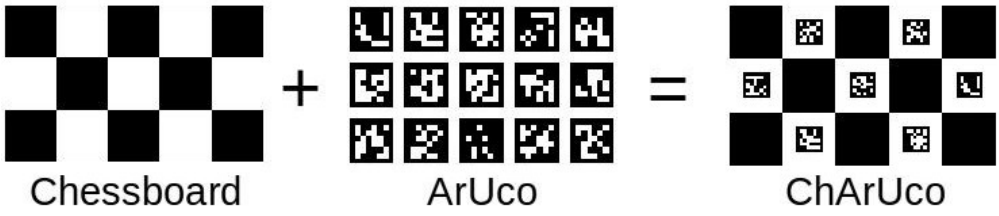

Рис. 7: Маркеры ChaAruCo

камера может слегка менять свое положение от запуска к запуску — штатив можно случайно задеть. Поэтому необходимо каждый раз заново четко идентифицировать рабочую область. Для этого можно закрепить на ней четко выделяющиеся и узнаваемые метки. Именно для этой цели мы еще раз используем метки ChAruCo. Именно четыре вариации этих меток закреплены по краям рабочей области, и при каждом запуске алгоритм их ищет и находит (рис. 9).

Поскольку рабочая область прямоугольная, то и изображение рабочей области, на котором производится поиск вырезаемых элементов, должно быть строго прямоугольным. Для этого к изображению применяется перспективное преобразование, переводящее опорные точки найденных четырех меток в углы прямоугольника:

$$[t_i x'_i, t_i y'_i, t_i]^T = M\,[x_i, y_i, 1]^T, i = 0, 1, 2, 3.$$

Здесь M – матрица преобразования, а разные i соответствуют четырем

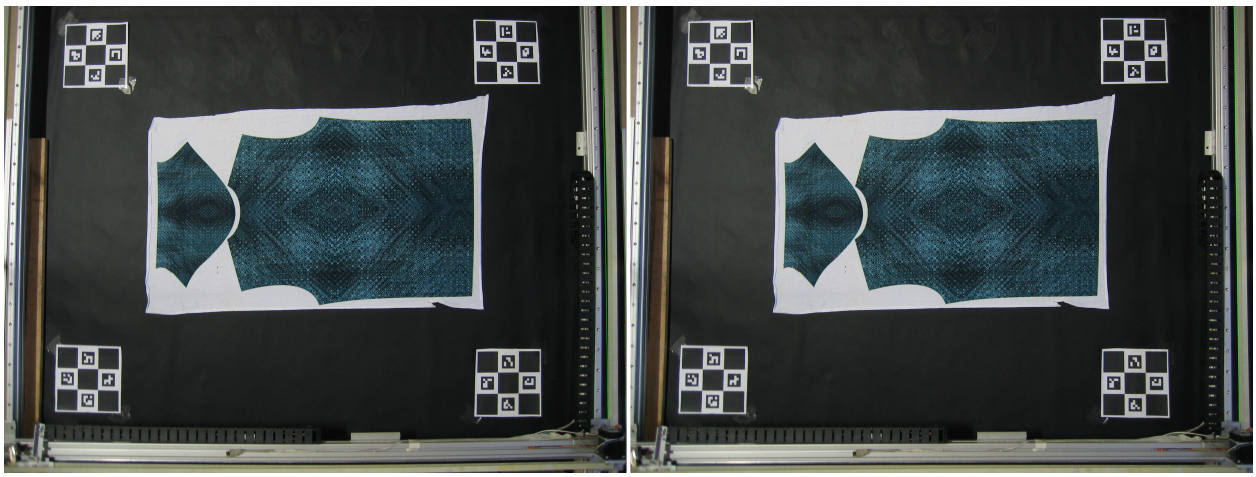

Рис. 8: Оригинально изображение (слева) и после устранения дисторсий (справа)

разным точкам. Если обозначить верхнюю левую, верхнюю правую, нижнюю правую, нижнюю левую точки через x_0, x_1, x_2, x_3 соответственно, то ширина нового изображения будет равна $max(||x_0–x_1||, ||x_2–x_3||)$, а высота $max(||x_0–x_3||, ||x_1–x_2||)$. В результате предпринятых действий мы имеем прямоугольное приближение рабочей области, которое позволит нам найти контуры объектов с нужной точностью (рис. 10).

Далее происходит работа по поиску контуров вырезаемых объектов. Сначала необходимо найти внешние контуры всех кусков ткани, расположенных на рабочей области.

Для того чтобы предотвратить негативное воздействие разного рода соринок, предварительно необходимо размыть изображение. Для этого производится двумерная свертка изображения с ядром Гаусса:

$$I'(i,j) = \sum_{l=-n}^{n} \sum_{k=-m}^{m} I(i+l, j+k) \frac{1}{2\pi\sigma^2} e^{-\frac{l^2+k^2}{2\sigma^2}},$$

где I — исходное изображение, I' – размытое изображение.

Затем производится обнаружение границ при помощи алгоритма Canny [9]. Алгоритм включает в себя четыре этапа:

1. Гауссово размытие изображения.

2. Свертка изображения с фильтром Собеля в вертикальном (G_x)

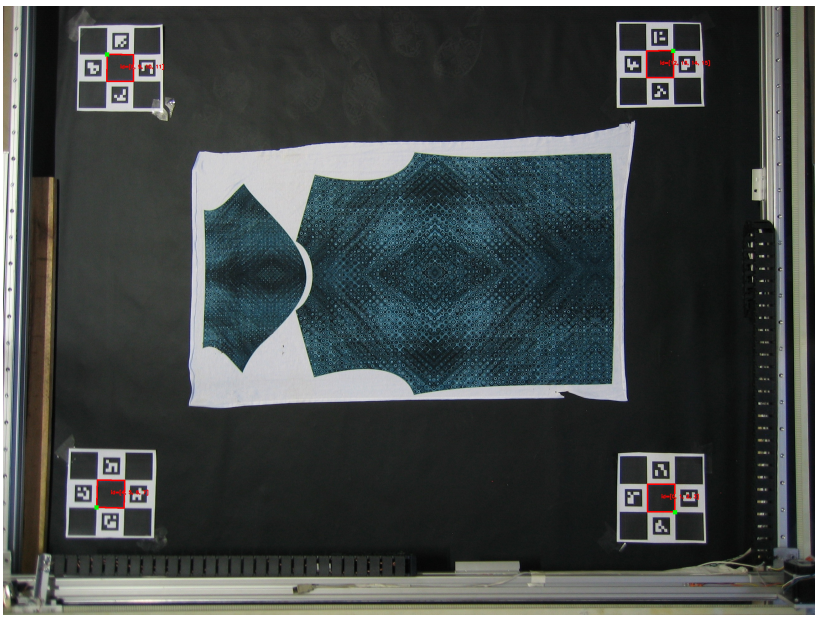

Рис. 9: Найденные метки

и горизонтальном направлениях (G_y), нахождение модуля ($G = \sqrt{G_x^2 + G_y^2}$) и направления градиента ($\theta = tan^{-1}(\frac{G_y}{G_x})$).

3. Подавление немаксимумов — оставление только тех точек, которые являются максимумами в направлениях градиента и антиградиента.

4. Процедура гистерезиса. Задаются два пороговых значения – нижнее v_0 и верхнее v_1. Утверждается, что точки, значение в которых больше, чем v_1, точно являются границами, а те, в которых меньше, чем v_0, точно не являются, и потому отбрасываются. Точки, которые лежат между двух значений, фильтруются по условию их соединенности с точками, лежащими выше и ниже порога. Если они соединены с точками, лежащими выше порога, то они принимаются алгоритмом, иначе отбрасываются.

В результате работы алгоритма имеем бинарное изображение, представленное на рис. 11.

Следующий шаг — нахождение контуров кусков ткани, лежащих на рабочей области. Для этого используется алгоритм, описанный в [28]. Он построчно итерируется по изображению, находит пиксели, опреде-

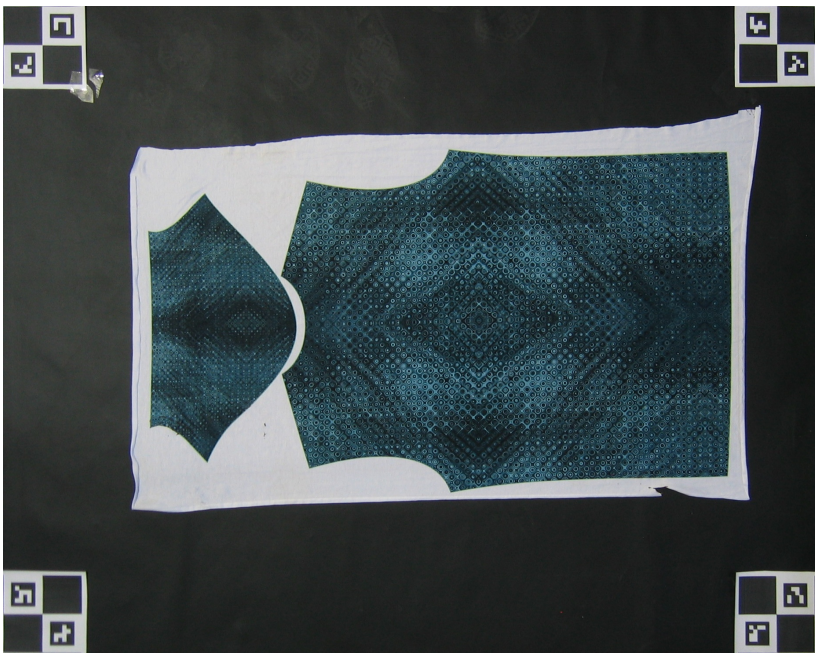

Рис. 10: Прямоугольное изображение

ляемые им как граничные, определяет тип границы, к которой они принадлежат, затем следует вдоль границы, помечая граничные пиксели разными числами для разных границ. Вдобавок к этому алгоритм строит дерево, определяя вложенность контуров друг в друга. Пусть алгоритм в результате своей работы выдал список найденных контуров. Необходимо отсортировать этот список по площади и столько элементов списка, сколько кусков ткани находится на рабочей области, либо просто взять контуры, площадь которых превышает заданный порог. Результат работы алгоритма можно видеть на рис. 12

Теперь требуется внутри найденных контуров найти контуры вырезаемых элементов. Для этого воспользуемся алгоритмом Flood Fill, а точнее его модификацией, применимой конкретно к данной задаче. Псведокод алгоритма приведен в листинге 1. Он получает на вход следующие параметры:

1. изображение;

2. бинарный двумерный массив, изначально заполненный нулями, в котором будут отмечены пиксели изображения, в которых алгоритм был запущен

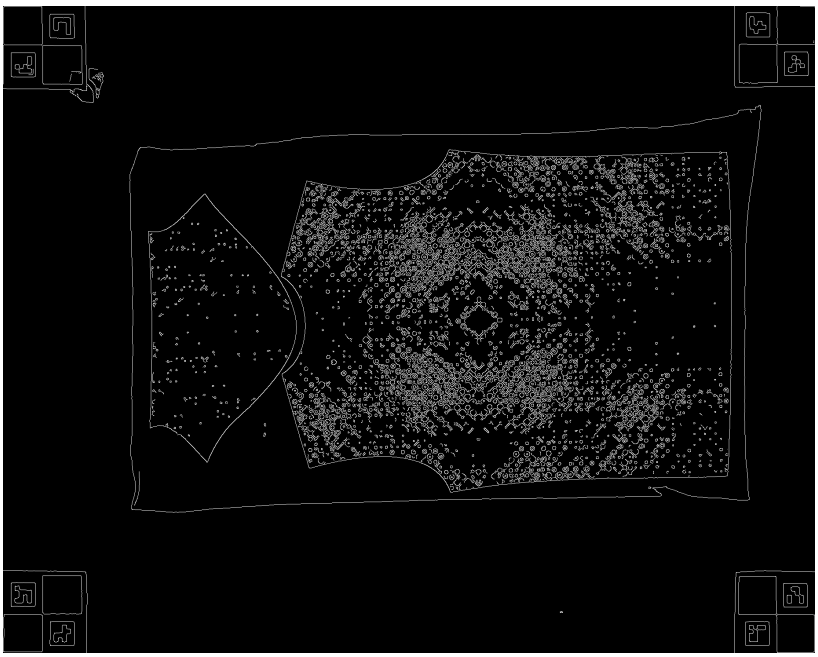

Рис. 11: Результат работы алгоритма Canny

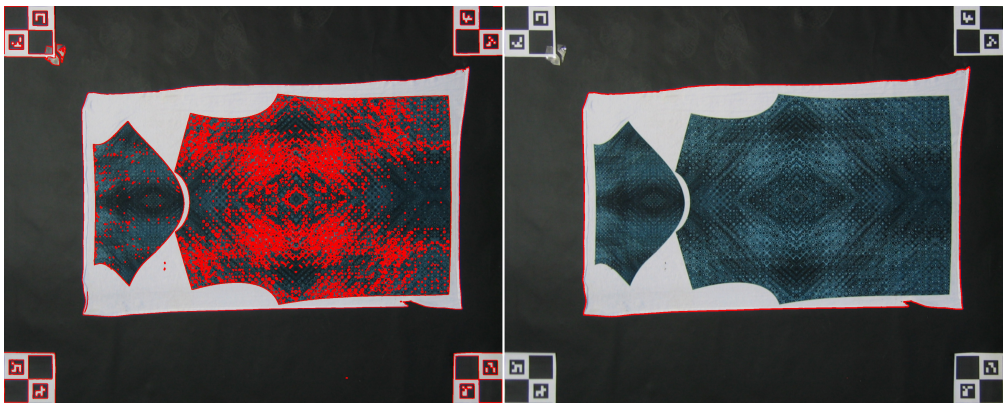

Рис. 12: Все найденные конутры (слева), искомый контур (справа)

3. координаты текущего пикселя изображения
4. два пороговых значения, нижнее и верхнее
5. опорный пиксель изображения, из которого требуется изначально запустить алгоритм

В самом начале алгоритма проверяется условие выхода из рекурсии. Алгоритм не продолжает свою работу, если уже запускался в текущем пикселе. В противном случае проверяется условие, что цвет текущего пикселя не отклоняется по каждому из каналов от цвета опорного пикселя более чем на low в нижнюю сторону и более чем на $high$ в верхнюю.

Если это так, то путем установки положительного значения в соответствующей ячейке массива *visited* алгоритм помечает текущий пиксель как пройденный и запускает себя из четырех соседних пикселей текущего. В результате работы алгоритма в массиве *visited* будут отмечена связная область, содержащая опорный пиксель, все остальные пиксели которой будут мало отличаться по цвету от опорного.

Алгоритм 1 $FloodFill(image, visited, x, y, low, high, x_{seed}, y_{seed})$

if $visited(x, y) == 1$ **then**
  return
$visited(x, y) = 1$
if $image(x, y)_i \in [image(x_{seed}, y_{seed})_i - low_i, image(x_{seed}, y_{seed})_i + high_i], i \in \{R, G, B\}$ **then**
  $FloodFill(image, visited, mask, x + 1, y, low, high, x_{seed}, y_{seed})$
  $FloodFill(image, visited, mask, x, y + 1, low, high, x_{seed}, y_{seed})$
  $FloodFill(image, visited, mask, x - 1, y, low, high, x_{seed}, y_{seed})$
  $FloodFill(image, visited, mask, x, y - 1, low, high, x_{seed}, y_{seed})$

Для того чтобы найти контуры вырезаемых элементов, необходимо найти связные области белого цвета, находящиеся между ними и внешними контурами кусков ткани. Для этого требуется запускать описанный алгоритм из точки, являющейся внутренней по отношению к найденному контуру куска ткани, но не являющейся внутренней по отношению к вырезаемым элементам. Для того чтобы найти такие точки, создадим бинарное изображение, в котором контуры кусков ткани, найденные на прошлом этапе, будут залиты белым цветом, а всё, что находится вне их — черным (рис. 13, слева). Затем произведем несколько итераций операции эрозии [16] над этим бинарным изображением со структурным элементом в виде эллипса размером $5x5$ и обозначим полученное изображение словом *stencil*. Эта позволит нам «сузить» белые области, и, таким образом, любая граничная точка этих «суженных» белых областей будет являться именно той точкой, из которой необходимо запустить алгоритм Flood Fill. Для этого снова воспользуемся алгоритмом поиска контуров [28], уже упомянутым и использованным ранее, и возьмем по одной точке с каждого контура. На рис. 13 справа показан такой контур, нарисованный на оригинальном изображении.

Создадим бинарное, полностью черное изображение, являющееся параметром *visited* алгоритма 1. Запустим этот алгоритм из найденных точек. В результате его работы массив *visited* будет иметь вид, показанный на рис 14 слева. Обозначим буквой S множество пикселей изображений *stencil*, равных 1. Обозначим буквой V множество пикселей изображения *visited*, равных 1. Вычислим изображение $result = S^c|V$ (определения операций см. в [16]), каждый пиксель которого будет равен 1, если соответствующий пиксель изображения *stencil* равен 0 или соответствующий пиксель изображения *mask* равен 1. Оно будет иметь вид, показанный на рис 14 справа. Для того чтобы окончательно получить контуры вырезаемых элементов, нужно еще раз воспользоваться алгоритмом поиска контуров, уже на изображении *result*. Результат приведен на рис. 15 слева.

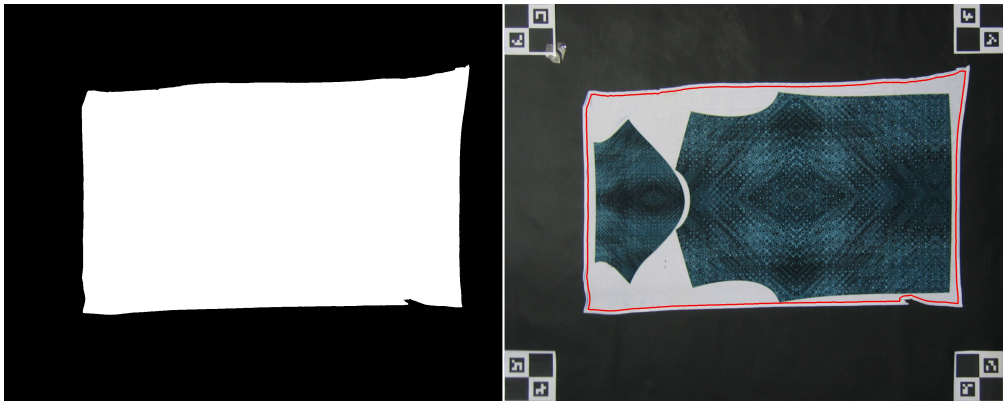

Рис. 13: Промежуточное изображение (слева) и «эродированный» контур (справа)

Следующая проблема заключается в том, что полученные контуры представляются слишком большим набором точек, что негативно сказывается на работе станка — он начинает двигаться весьма нестабильно, что может плохо сказаться на качестве резки при работе с мощным лазером. Поэтому необходимо «упростить» контуры, воспользовавшись одним известным алгоритмом [10]. Суть алгоритма состоит в том, чтобы по данной ломаной, аппроксимирующей кривую, построить ломаную с меньшим числом точек. Алгоритм определяет расхождение, которое вычисляется по максимальному расстоянию между исходной и упрощенной кривыми. Упрощенная кривая состоит из подмножества

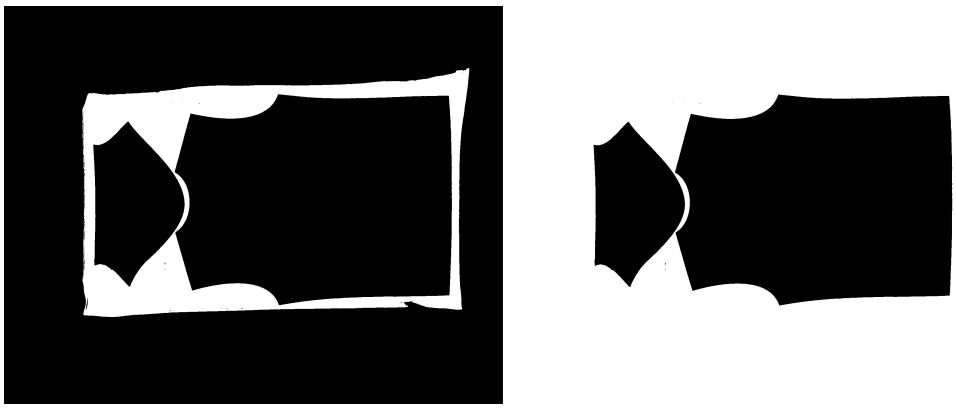

Рис. 14: Результат работы алгоритма Flood Fill (слева) и промежуточное изображение (справа)

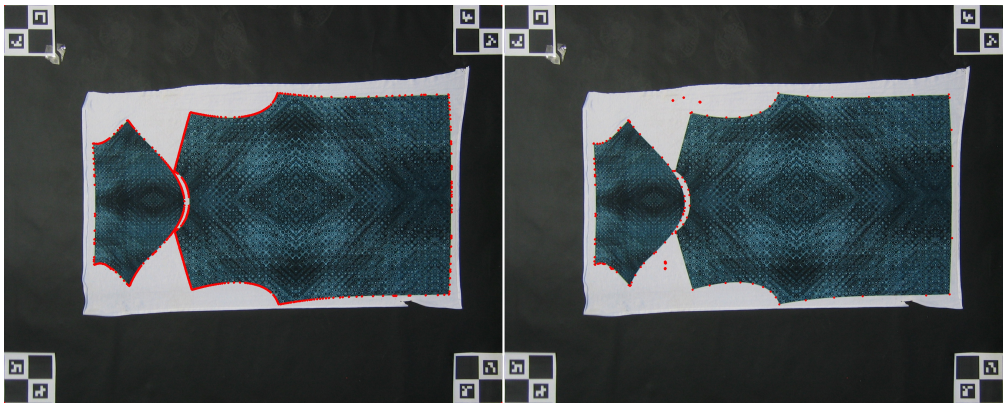

Рис. 15: Исходный контур (слева) и «упрощенный» контур (справа)

точек, которые определяются из исходной кривой. Результат работы алгоритма можно видеть на рис. 15 справа.

На данном этапе точки контура имеют масштаб, отличающийся от масштаба рабочей области. Необходимо умножить точки контура на сжимающие коэффициенты, являющиеся отношением величины рабочей области в мм по каждой оси к величине соответствующей стороны изображения в пикселях. В идеальной ситуации эти коэффициенты должны быть равны, на практике приемлемо, чтобы они совпали хотя бы до нескольких знаков после запятой.

Теперь имеется необходимая информация для того чтобы приступить к резке материала. Аппаратная часть станка управляется контроллером Arduino Uno с прошивкой Grbl [17]. Эта прошивка является программой, которая запускается сразу же при подаче питания на кон-

троллер. Она способна воспринимать команды специального языка программирования G-code [14], созданного в 1960-х годах для управления устройствами с ЧПУ. Основные используемые команды для управления станком:

1. G01 – включает линейную интерполяцию. Вместе с ней задаются начальная и конечная точки, и контроллер автоматически вычисляет (интерполирует) промежуточные точки, вычисляет необходимые угловые скорости для моторов.

2. G90 – включает абсолютное позиционирование

3. M3, M5 – включает и выключает лазер

4. S – устанавливает скорость

5. X, Y – задает координаты точек рабочей области

Программный код для воспроизведения данной работы можно найти по ссылке `https://github.com/majestic905/laser_cnc`.

Заключение

В данной работе рассмотрена и с относительным успехом решена одна частная задача, сочетающая в себе работу с современным контроллером с одной стороны и алгоритмами компьтерного зрения и обработки изображений с другой, а именно построение контура объекта по набору изображений, в рамках создания аппаратно-программного комплекса по созданию станка для лазерного раскроя швейных материалов.

Станок был протестирован на выборке из 30 кусков ткани, и показал приемлемый результат в 80% случаев.

Отличительной чертой используемого подхода стало использование камеры для определения вырезаемых элементов. Предложенный алгоритм был протестирован, показал на практике хороший результат и удовлетворил заказчика.

Плюсами алгоритма и написанного программного обеспечения можно отметить следующее. Алгоритм довольно прост, что позволяет его быстро реализовать в виде программы, а также обеспечивает его быструю работу даже при малоэффективной реализации. Конкретная программа, написанная в ходе данной работы, имеет модульную структуру, и один из модулей отвечает за непосредственное выделение объектов для вырезания. Он принимает на вход прямоугольное изображение рабочей области, а выходом его является массив контуров. Также программа написана на популярном языке программирования с относительно низким порогом вхождения, что позволяет человеку, который захочет как-то улучшить или модифицировать ее, сделать это с меньшими трудозатратами. Из плюсов станка можно также отметить простоту устройства и сборки — такой станок легко можно собрать в одиночку.

Из минусов данной работы, а точнее ее аппаратной части, можно отметить не вполне удачный выбор лазера. Также при использовании лазера в качестве режущего устройства требуется наличие специального стола, вытяжки, хорошо проветриваемого помещения. Помимо этого, работа с любым лазерным устройством подразумевает повышенные

требования к технике безопасности — как минимум надевание специальных защитных очков перед началом работы.

Перспективными направлениями развития проекта, как в рамках требований данной задачи, так и уже в других проектах, можно отметить следующее:

1. увеличение рабочей области при необходимости
2. установка камеры с более высоким разрешением для обеспечения субмиллиметровой точности
3. отмечавшуюся модульную структуру программы вместе с использованием других алгоритмов, методов и подходов компьютерного зрения и обработки изображений можно использовать для нахождения и вырезания более сложных фигур, причем не только на ткани

Стоимость деталей станка оценивается следующим образом: двигатели — 3920 р $\times$ 3 шт, драйвера — 6930 р $\times$ 2 шт, блок питания — 5390 р, контроллера Arduino Uno – 470 р, камера Canon PowerShot S50 – 700 р, лазер — 10600 р, алюминиевые профили — примерно 5500 р. В сумме получается примерно 48280 р с учетом нынешних курсов доллара и евро.

Список литературы

[1] 450nm 5W Laser Engraving Module Blue Light Marking Engraver With TTL Modulation. — 2018. — URL: https://www.banggood.com/450nm-5W-Laser-Engraving-Module-Blue-Light-Marking-Engraver-Wit html?currency=EUR&utm_source=criteo&utm_medium=cpc&utm_content=all&utm_campaign=m-electronics-EU-English&stayold=1&cur_warehouse=USA (online; accessed: 25.05.2018).

[2] A. Basu S. Licardie. Alternative models for fish-eye lenses // Pattern Recognition Letters. — 1995. — Vol. 16, no. 4. — P. 443–441.

[3] Arduino Uno Rev3. — 2018. — URL: https://store.arduino.cc/arduino-uno-rev3 (online; accessed: 25.05.2018).

[4] Automatic generation and detection of highly reliable fiducial markers under occlusion / S. Garrido-Jurado, R. Muñoz Salinas, F.J. Madrid-Cuevas, M.J. Marín-Jiménez // Pattern Recognition. — 2014. — Vol. 47, no. 6. — P. 2280 – 2292. — URL: http://www.sciencedirect.com/science/article/pii/S0031320314000235.

[5] Brown D. C. Decentering distortion of lenses // Photogrammetric Engineering and Remote Sensing. — 1966.

[6] Brown D. C. Close-range camera calibration // Photogrammetric Engineering and Remote Sensing. — Vol. 8. — 1971. — P. 855–866.

[7] C. McGlone E. Mikhail J. Bethel. Manual of Photogrammetry, fifth ed. — American Society of Photogrammetry and Remote Sensing, 2004.

[8] Camera Calibration Toolbox for Matlab. — 2018. — URL: http://www.vision.caltech.edu/bouguetj/calib_doc/ (online; accessed: 25.05.2018).

[9] Canny J. A Computational Approach to Edge Detection // IEEE

Transactions on Pattern Analysis and Machine Intelligence. — 1986. — Vol. PAMI-8, no. 6. — P. 679–698.

[10] Douglas David H., Peucker Thomas K. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature // The Canadian Cartographer. — 1973. — Vol. 10, no. 2. — P. 112–122.

[11] F. Devernay O. Faugeras. Straight lines have to be straight // Machine Vision Applications. — 2001. — Vol. 13, no. 1. — P. 14–24.

[12] Fitzgibbon A. Simultaneous Linear Estimation of Multiple view Geometry and lens distortion // in Proceedings of the Conference on Computer Vision and Pattern Recognition. — 2001.

[13] Fryer J. G., Brown D. C. Lens Distortion for Close-Range Photogrammetry // Photogrammetric Engineering and Remote Sensing. — 1986. — Vol. 52, no. 1. — P. 51–58.

[14] G-code - Wikipedia. — 2018. — URL: `https://en.wikipedia.org/wiki/G-code` (online; accessed: 25.05.2018).

[15] Generation of fiducial marker dictionaries using mixed integer linear programming / S. Garrido-Jurado, R. Muñoz Salinas, F.J. Madrid-Cuevas, R. Medina-Carnicer // Pattern Recognition. — 2016. — Vol. 51. — P. 481 – 491. — URL: `http://www.sciencedirect.com/science/article/pii/S0031320315003544`.

[16] Gonzalez Rafael C., Woods Richard E. Digital Image Processing (3rd Edition). — Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 2006. — ISBN: 013168728X.

[17] Home - gnea/grbl Wiki. — 2018. — URL: `https://github.com/gnea/grbl/wikigrbl` (online; accessed: 25.05.2018).

[18] J. Mallon P.F. Whelan. Precise radial un-distortion of images // in Proceedings of the 17th International Conference on Pattern Recognition. — 2004.

[19] J. Weng P. Cohen M. Herniou. Camera calibration with distortion models and accuracy evaluation // IEEE Transactions Pattern Analysis and Machine Intelligence. — 1992. — Vol. 14, no. 10. — P. 965–980.

[20] L. Ma Y.Q. Chen K.L. Moore. Flexible camera calibration using a new analytical radial undistortion formula with application to mobile robot localization // in Proceedings of IEEE International Symposium on Intelligent Control / IEEE. — 2003.

[21] Leadshine Technology Co, Ltd. — 2018. — URL: http://www.leadshine.com/productdetail.aspx?model=57HS13 (online; accessed: 25.05.2018).

[22] Leadshine Technology Co, Ltd. — 2018. — URL: http://www.leadshine.com/productdetail.aspx?model=DM556 (online; accessed: 25.05.2018).

[23] Leadshine Technology Co, Ltd. — 2018. — URL: http://www.leadshine.com/productdetail.aspx?model=SPS407 (online; accessed: 25.05.2018).

[24] PowerShot S50 // Canon Camera Museum. — 2018. — URL: http://global.canon/en/c-museum/product/dcc482.html (online; accessed: 25.05.2018).

[25] Ricolfe-Viala Carlos, Sanchez-Salmeron Antonio-Jose. Lens distortion models evaluation // Appl. Opt. — 2010. — Oct. — Vol. 49, no. 30. — P. 5914–5928. — URL: http://ao.osa.org/abstract.cfm?URI=ao-49-30-5914.

[26] S. Shah J.K. Aggarwal. Intrinsic parameter calibration procedure for a (high distortion) fish-eye lens camera with distortion model and accuracy estimation // Pattern Recognition. — 1996. — Vol. 29, no. 11. — P. 1775–1778.

[27] S.S. Beauchemin R. Bajcsy. Modelling and removing radial and tangential distortions in spherical lenses, Multi Image Analysis // Lecture Notes in Computer Science. — 2001. — Vol. 2023. — P. 1–21.

[28] Suzuki Satoshi, Abe Keiichi. Topological structural analysis of digitized binary images by border following // Computer Vision, Graphics, and Image Processing. — 1985. — Vol. 30, no. 1. — P. 32–46.

[29] Tsai R. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-self TV camera lenses // IEEE Journal of Robotics and Automation. — 1997. — Vol. 4. — P. 323–344.

[30] Zhang Z. A flexible new technique for camera calibration // IEEE Transactions on Pattern Analysis and Machine Intelligence. — 2000. — Vol. 22, no. 11. — P. 1330–1334.

[31] gPhoto - Projects :: libgphoto2. — 2018. — URL: `http://www.gphoto.org/proj/libgphoto2/` (online; accessed: 25.05.2018).